CSE 543 - Computer Security

Lecture 6 - Authentication
September 21, 2006

URL: http://www.cse.psu.edu/~tjaeger/cse543-f06/
Project Background and Related Work

• Due 10/10

• Questions to Answer:
  – What is the technical problem?
  – What has been done to solve it in the past?
  – Why has no one solved it yet?

• On project assignments page (~tjaeger/project_assigns.html)
  – More resources to investigate answers
  – Often tip of the iceberg
    • References in papers
    • Systems mentioned

• Divide up search for information and answer the questions above
What is Authentication?

• Short answer: establishes identity
  – Answers the question: To whom am I speaking?

• Long answer: evaluates the authenticity of identity proving credentials
  – Credential – is proof of identity
  – Evaluation – process that assesses the correctness of the association between credential and claimed identity
    • for some purpose
    • under some policy
Why authentication?

• Well, we live in a world of rights, permissions, and duties?
  – Authentication establishes our identity so that we can obtain the set of rights
  – E.g., we establish our identity with Tiffany’s by providing a valid credit card which gives us rights to purchase goods ~ physical authentication system

• Q: How does this relate to security?
Why authentication (cont.)?

• Same in online world, just different constraints
  – Vendor/customer are not physically co-located, so we must find other ways of providing identity
    • e.g., by providing credit card *number* ~ electronic authentication system
  – Risks (for customer and vendor) are different
    • Q: How so?

• *Computer security is crucially dependent on the proper design, management, and application of authentication systems.*
What is Identity?

• That which gives you access … which is largely determined by context
  – We all have lots of identities
  – Pseudo-identities

• Really, determined by who is evaluating credential
  – Driver’s License, Passport, SSN prove …
  – Credit cards prove …
  – Signature proves …
  – Password proves …
  – Voice proves …

• Exercise: Give an example of bad mapping between identity and the purpose for which it was used.
Credentials

- … are evidence used to prove identity
- Credentials can be
  - Something I am
  - Something I have
  - Something I know
Something you know …

- Passport number, mother’s maiden name, last 4 digits of your social security, credit card number

- Passwords and pass-phrases
  - Note: passwords are generally pretty weak
    - University of Michigan: 5% of passwords were goblue
    - Passwords used in more than one place
  - Not just because bad ones selected: If you can remember it, then a computer can guess it
    - Computers can often guess very quickly
    - Easy to mount offline attacks
    - Easy countermeasures for online attacks
Something your have …

• Tokens (transponders, …)
  – Speedpass, EZ-pass

• Smartcards

• Digital Certificates (used by Websites to authenticate themselves to customers)
  – More on this later …
Something your are ...

• Biometrics measure some physical characteristic
  – Fingerprint, face recognition, retina scanners, voice, signature, DNA
  – Can be extremely accurate and fast
  – Active biometrics authenticate
  – Passive biometrics recognize

• What is the fundamental problem?
  – Revocation – lost fingerprint?
  – Great for physical security, generally not feasible for online systems
Web Authentication

• Authentication is a bi-directional process
  – Client
  – Server
  – Mutual authentication

• Several standard authentication tools
  – Basic (client)
  – Digest (client)
  – Secure Socket Layer (server, mutual)
  – Cookies (indirect, persistent)

• Q: Are cookies good credentials?
How Basic Authentication Works …

CLIENT

GET /protected/index.html HTTP/1.0

HTTP/1.0 401 Unauthorized
WWW-Authenticate: Basic realm="Private"

CLIENT

GET /protected/index.html HTTP/1.0
Authorization: Basic JA87JKAs3NbBDs

CLIENT
Setting up Basic auth in Apache

• File in directory to protect (.htaccess)

```bash
AuthType Basic
AuthName Patrick's directions (User ID=pdmcdan)"
AuthUserFile /usr/pdmcdan/www/etc/.htpw1
AuthGroupFile /dev/null
require valid-user
```

• In /usr/pdmcdan/www/etc/.htpw1
  
  pdmcdan:17fweqjyzmno

generated using htpasswd program

• Can use different .htaccess files for different directories
Basic Authentication Problems

- Passwords easy to intercept
- Passwords easy to guess
  - Just base-64 encoded
- Passwords easy to share
- No server authentication
  - Easy to fool client into sending password to malicious server
- One intercepted password gives eavesdropper access to many documents
Digest Authentication

GET /protected/index.html HTTP/1.1

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest
realm="Private" nonce="98bdc1f9f017.."

GET /protected/index.html HTTP/1.1
Authorization: Digest
username="lstein" realm="Private"
nonce="98bdc1f9f017.." response="5ccc069c4.."
Challenge and Response

• Challenge ("nonce"): *any changing string*
  - e.g. MD5(IP address:timestamp:server secret)

• Response: *challenge hashed with user’s name & password*
  - MD5(MD5(name:realm:password):nonce:MD5(request))

• Server-specific implementation options
  • One-time nonces
  • Time-stamped nonces
  • Method authentication digests
Advantages of Digest over Basic

- Cleartext password never transmitted across network
- Cleartext password never stored on server
- Replay attacks difficult
- Intercepted response only valid for a single URL
- Shared disadvantages
  - Vulnerable to man-in-the-middle attacks
  - Document itself can be sniffed
Kerberos

• History: from UNIX to Networks (late 80s)
  – Solves: password eavesdropping
  – Online authentication
    • Variant of Needham-Schroeder protocol
  – Easy application integration API
  – First *single sign-on system* (SSO)
  – Genesis: rsh, rcp
    • authentication via assertion

• Most widely used (non-web) centralized password system in existence (and lately only ..)
• Now: part of Windows 2K, XP network authentication
  – Windows authentication was a joke.
An aside …

- **Authentication**
  - Assessing identity of users
  - By using credentials …

- **Authorization**
  - Determining if users have the right to perform requested action (e.g., write a file, query a database, etc.)

- **Kerberos authenticates users, but does not perform any authorization functions …**
  - … beyond identify user as part of Realm
  - Typically done by application.

- **Q: Do you use any “Kerberized” programs?**
  - How do you know?
The setup ...

**The players**
- Principal - person being authenticated
- Service (verifier) - entity requiring authentication (e.g., AFS)
- Key Distribution Center (KDC)
  - Trusted third party for key distribution
  - Each principal and service has a Kerberos password known to KDC, which is munged to make a password $ke$, e.g., $k^A$
- Ticket granting server
  - Server granting transient authentication

**The objectives**
- Authenticate Alice (Principal) to Bob (Service)
- Negotiate a symmetric (secret) session key $k^{AB}$
The protocol

• A two-phase process
  – User authentication/obtain session key (and ticket granting ticket) key from Key Distribution Center
  – Authenticate Service/obtain session key for communication with service

• Setup
  – Every user and service get certified and assigns password
A Kerberos Ticket

- A kerberos ticket is a token that …
  - Alice is the only one that can open it
  - Contains a session key for Alice/Bob ($K_{AB}$)
  - Contains *inside it* a token that can only be opened by Bob

- Bob’s Ticket contains
  - Alice’s identity
  - The session key ($K_{AB}$)

- Q: What if issuing service is not trusted?
The protocol (obtaining a TGT)

- $T_{\text{exp}}$ - time of expiration
- $n$ - nonce (random, one-use value: e.g., timestamp)

$$\begin{align*}
1 & \quad [A, TGS, T_{\text{exp}}, n] \\
2 & \quad E(k^A, [k^{A,TGS}, TGS, T_{\text{exp}}, n]), E(K^{TGS}, [A, k^{A,TGS}, T_{\text{exp}}]),
\end{align*}$$

TGT
The protocol (performing authentication)

1. Alice
   - $E(k_{A,TGS}, [B, \text{Time}_{\text{exp}}, n])$, $E(K_{TGS}, [A, k_{A,TGS}, \text{Time}_{\text{exp}}])$

2. TGS
   - $E(k_{A,B}, [B, \text{Time}_{\text{exp}}, n])$, $E(k_{B}, [A, k_{A,B}, \text{Time}_{\text{exp}}])$

3. Bob
   - $E(k_{A,B}, [A, \text{Time}_{\text{exp}}, n])$, $E(k_{B}, [A, k_{A,B}, \text{Time}_{\text{exp}}])$

Authenticator
In class

- Work in binary numbers
- Cipher = XOR
  - key=0111000
  - plaintext =10101111
  - ciphertext $E(k,p) = 01110000 \ XOR \ 10101111 = 11011111$
  - plaintext = 11011111 XOR 01110000 = 10101111

- Groups of 4
  - Alice (principal)
  - Bob (service)
  - Key distribution center (KDC)
  - Ticket granting server (TGS)
Protocol (setup and phase 1)

- Each Alice, Bob, and TGS *secretly* negotiate a 8-bit key with the KDC
  - Only Alice and KDC know $k^{A,KDC}$
  - Only Bob, KDC, and TGS know $k^{B,KDC}$
    - Note: KDC exposes Bob’s key to TGS
  - Only TGS and KDC know $k^{KDC,TGS}$

- Phase 1 (obtaining a ticket-granting ticket)
  1. $A \rightarrow \text{KDC} : A$
  2. $\text{KDC} \rightarrow A : E(k^{A,KDC},[k^{A,TGS}]), E(k^{KDC,TGS},[k^{A,TGS}])$
    - where $k^{A,TGS}$ is randomly selected by KDC
Protocol Phase 2 and communication

• Phase 2 (obtaining a service ticket)
  1. A -> TGS : B, E(k^{KDC,TGS}, [k^{A,TGS}])
  2. TGS -> A : E(k^{A,TGS}, [k^{A,B}]), E(k^{B,KDC}, [k^{A,B}])
      - where k^{A,B} is randomly selected by TGS
    – A -> B : E(k^{B,KDC}, [k^{A,B}])

• Communications
  1. A -> B : E(k^{A,B}, [01011010])
  2. B -> A : E(k^{A,B}, [10010110])
Cross-Realm Kerberos

• Extend philosophy to more servers
  – Obtain ticket from TGS for foreign Realm
  – Supply to TGS of foreign Realm
  – Rinse and repeat as necessary

• “There is no problem so hard in computer science that it cannot be solved by another layer of indirection.”
  – David Wheeler, Cambridge University (circa 1950)
Kerberos Reality

• V4 was supposed to be replaced by V5
  – But wasn’t because interface was ugly, complicated, and encoding was infuriating

• Assumes *trusted path* between user and Kerberos

• Widely used in UNIX domains

• Robust and stable implementation

• *Problem*: trust ain’t transitive, so not so good for large collections of autonomous enterprises